CBIOS2.PRN

OLD

```
                 *****************************************************************
                 *                                                               *
                 * CBIOS FOR CP/M VER 2.2 FOR DISK JOCKEY 2D CONTROLLER (ALL     *
                 * REVS). HANDLES DISKETTES WITH SECTOR SIZES OF 128 BYTES       *
                 * SINGLE DENSITY, 256, 512, 1Ø24 BYTES DOUBLE DENSITY.          *
                 *                                                               *
                 * WRITTEN BY BOBBY DALE GIFFORD.                                *
                 * 9/1/79                                                        *
                 *                                                               *
                 * CUSTOMIZED BY JAY O'BRIEN.                                    *
                 * 4/12/81                                                       *
                 *                                                               *
                 * DISK MAP OF SECTORS USED BY COLD BOOT, WARM BOOT, FIRMWARE,   *
                 * AND CP/M:                                                     *
                 *                                                               *
                 * TRK Ø SEC  1 = FIRST SECTOR OF COLD BOOT.             E7ØØH   *
                 *     Ø      2 = COLD BOOT 256.                           8ØH   *
                 *     Ø      3 = COLD BOOT 512.                           8ØH   *
                 *     Ø      4 = COLD BOOT 1Ø24.                          8ØH   *
                 *     Ø      5 = WARM BOOT 256.                           8ØH   *
                 *     Ø      6 = WARM BOOT 512.                           8ØH   *
                 *     Ø      7 = WARM BOOT 1Ø24.                          8ØH   *
                 *     Ø      8 = COLD/WARM BOOT.                        32ØØH   *
                 *     Ø      9 = FIRMWARE.                              E4ØØH   *
                 *     Ø     1Ø = FIRMWARE+8ØH.                          E48ØH   *
                 *     Ø     11 = FIRMWARE+1ØØH                          E5ØØH   *
                 *     Ø     12 = FIRMWARE+18ØH.                         E58ØH   *
                 *     Ø     13 = FIRMWARE+2ØØH.                         E6ØØH   *
                 *     Ø     14 = FIRMWARE+28ØH.                         E68ØH   *
                 *     Ø     15 = FIRMWARE+3ØØH.                         E7ØØH   *
                 *     Ø     16 = FIRMWARE+38ØH.                         E78ØH   *
                 *     Ø     17 = CCP.                                   2DØØH   *
                 *     Ø     1Ø = CCP+8ØH.                               2D8ØH   *
                 *     Ø     12 = CCP+1ØØH.                              2EØØH   *
                 *     Ø     14 = CCP+18ØH.                              2E8ØH   *
                 *     Ø     16 = CCP+2ØØH.                              2FØØH   *
                 *     Ø     18 = CCP+28ØH.                              2F8ØH   *
                 *     Ø     2Ø = CCP+3ØØH.                              3ØØØH   *
                 *     Ø     22 = CCP+38ØH.                              3Ø8ØH   *
                 *     Ø     24 = CCP+4ØØH.                              31ØØH   *
                 *     Ø     26 = CCP+48ØH.                              318ØH   *
                 *     1        = REST OF CP/M.                    32ØØH-4FFFH   *
                 *                                                               *
                 *****************************************************************

                         TITLE   '*** Cbios For CP/M Ver. 2.2 ***'

                 *****************************************************************
                 *                                                               *
                 * THE FOLLOWING REVISION NUMBER IS IN REFERENCE TO THE CP/M     *
                 * 2.Ø CBIOS.                                                    *
                 *                                                               *
                 *****************************************************************

ØØ1E =           REVNUM  EQU     3Ø              ;CBIOS REVISION NUMBER
ØØ16 =           CPMREV  EQU     22              ;CP/M REVISION NUMBER
```

```
                    ****************************************************************
                    *                                                              *
                    * THE FOLLOWING EQUATES RELATE THE THINKER TOYS 2D CONTROLLER. *
                    * IF THE CONTROLLER IS NON STANDARD (0E000H) ONLY THE ORIGIN   *
                    * EQUATE NEED BE CHANGED. THIS VERSION OF THE CBIOS WILL WORK  *
                    * WITH 2D CONTROLLER BOARDS REV 0, 1, 3, 3.1, 4.               *
                    *                                                              *
                    ****************************************************************

E000 =              ORIGIN  EQU     0E000H
E400 =              DJRAM   EQU     ORIGIN+400H     ;DISK JOCKEY 2D RAM ADDRESS
E403 =              DJCIN   EQU     DJRAM+3H        ;DISK JOCKEY 2D CHARACTER INPUT ROUTINE
E406 =              DJCOUT  EQU     DJRAM+6H        ;DISK JOCKEY 2D CHARACTER OUTPUT ROUTINE
E409 =              DJHOME  EQU     DJRAM+9H        ;DISK JOCKEY 2D TRACK ZERO SEEK
E40C =              DJTRK   EQU     DJRAM+0CH       ;DISK JOCKEY 2D TRACK SEEK ROUTINE
E40F =              DJSEC   EQU     DJRAM+0FH       ;DISK JOCKEY 2D SET SECTOR ROUTINE
E412 =              DJDMA   EQU     DJRAM+012H      ;DISK JOCKEY 2D SET DMA ADDRESS
E415 =              DJREAD  EQU     DJRAM+15H       ;DISK JOCKEY 2D READ ROUTINE
E418 =              DJWRITE EQU     DJRAM+18H       ;DISK JOCKEY 2D WRITE ROUTINE
E41B =              DJSEL   EQU     DJRAM+1BH       ;DISK JOCKEY 2D SELECT DRIVE ROUTINE
E421 =              DJTSTAT EQU     DJRAM+21H       ;DISK JOCKEY 2D TERMINAL STATUS ROUTINE
E427 =              DJSTAT  EQU     DJRAM+27H       ;DISK JOCKEY 2D STATUS ROUTINE
E42A =              DJERR   EQU     DJRAM+2AH       ;DISK JOCKEY 2D ERROR, FLASH LED
E42D =              DJDEN   EQU     DJRAM+2DH       ;DISK JOCKEY 2D SET DENSITY ROUTINE
E430 =              DJSIDE  EQU     DJRAM+30H       ;DISK JOCKEY 2D SET SIDE ROUTINE

                    ****************************************************************
                    *                                                              *
                    * EQUATES FOR MY SYSTEM.  J.J. O'BRIEN                         *
                    *                                                              *
                    ****************************************************************

E800 =              MSDV    EQU     0E800H          ;VIDEO DRIVER FOR MSDV

                    ****************************************************************
                    *                                                              *
                    * CP/M SYSTEM EQUATES. IF RECONFIGURATION OF THE CP/M SYSTEM   *
                    * IS BEING DONE, THE CHANGES CAN BE MADE TO THE FOLLOWING      *
                    * EQUATES.                                                     *
                    *                                                              *
                    ****************************************************************

0038 =              MSIZE   EQU     56              ;MEMORY SIZE OF TARGET CP/M
9000 =              BIAS    EQU     (MSIZE-20)*1024 ;MEMORY OFFSET FROM 20K SYSTEM
BD00 =              CCP     EQU     2D00H+BIAS      ;CONSOLE COMMAND PROCESSOR
C500 =              BDOS    EQU     CCP+800H        ;BDOS ADDRESS
D300 =              BIOS    EQU     CCP+1600H       ;CBIOS ADDRESS
0004 =              CDISK   EQU     4               ;ADDRESS OF LAST LOGGED DISK
0080 =              BUFF    EQU     80H             ;DEFAULT BUFFER ADDRESS
0100 =              TPA     EQU     100H            ;TRANSIENT MEMORY
00C0 =              INTIOBY EQU     192             ;INITIAL IOBYTE
0003 =              IOBYTE  EQU     3               ;IOBYTE LOCATION
0000 =              WBOT    EQU     0               ;WARM BOOT JUMP ADDRESS
0005 =              ENTRY   EQU     5               ;BDOS ENTRY JUMP ADDRESS
```

```
                    ****************************************************************
                    *                                                              *
                    * THE FOLLOWING ARE INTERNAL CBIOS EQUATES. MOST ARE MISC.     *
                    * CONSTANTS.                                                   *
                    *                                                              *
                    ****************************************************************

ØØØA =              RETRIES EQU     1Ø               ;MAX RETRIES ON DISK I/O BEFORE ERROR
ØØØD =              ACR     EQU     ØDH              ;A CARRIAGE RETURN
ØØØA =              ALF     EQU     ØAH              ;A LINE FEED
ØØØ3 =              AETX    EQU     3                ;A ETX CHAR
ØØØ6 =              AACK    EQU     6                ;A ACK CHAR
ØØ19 =              CLEAR   EQU     19H              ;CLEAR SCREEN FOR MSDV
ØØØ4 =              MAXDISK EQU     4                ;MAXIMUM # OF DISK DRIVES
ØØØ8 =              DBLSID  EQU     8                ;SIDE BIT FROM CONTROLLER

                    ****************************************************************
                    *                                                              *
                    * THE JUMP TABLE BELOW MUST REMAIN IN THE SAME ORDER, THE      *
                    * ROUTINES MAY BE CHANGED, BUT THE FUNCTION EXECUTED MUST BE   *
                    * THE SAME.                                                    *
                    *                                                              *
                    ****************************************************************

D3ØØ                        ORG     BIOS             ;CBIOS STARTING ADDRESS

D3ØØ C3AØD3                 JMP     CBOOT            ;COLD BOOT ENTRY POINT
D3Ø3 C3FCD3         WBOOTE  JMP     WBOOT            ;WARM BOOT ENTRY POINT
D3Ø6 C34ØD6                 JMP     CONST            ;CONSOLE STATUS ROUTINE
D3Ø9 C34CD6                 JMP     CONIN            ;CONSOLE INPUT
D3ØC C361D6         COUT    JMP     CONOUT           ;CONSOLE OUTPUT
D3ØF C381D6                 JMP     LIST             ;LIST DEVICE OUTPUT
D312 C376D6                 JMP     PUNCH            ;PUNCH DEVICE OUTPUT
D315 C36CD6                 JMP     READER           ;READER DEVICE INPUT
D318 C39ØD4                 JMP     HOME             ;HOME DRIVE
D31B C3C6D4                 JMP     SETDRV           ;SELECT DISK
D31E C392D4                 JMP     SETTRK           ;SET TRACK
D321 C385D4                 JMP     SETSEC           ;SET SECTOR
D324 C38AD4                 JMP     SETDMA           ;SET DMA ADDRESS
D327 C369D5                 JMP     READ             ;READ THE DISK
D32A C362D5                 JMP     WRITE            ;WRITE THE DISK
D32D C38CD6                 JMP     LISTST           ;LIST DEVICE STATUS
D33Ø C397D4                 JMP     SECTRAN          ;SECTOR TRANSLATION
D333 C31BE4         DJDRV   JMP     DJSEL            ;HOOK FOR SINGLE.COM PROGRAM

                    ****************************************************************
                    *                                                              *
                    * SIGNON MESSAGE OUTPUT DURING COLD BOOT.                      *
                    *                                                              *
                    ****************************************************************

D336 ØDØAØA         PROMPT  DB      ACR,ALF,ALF
D339 35                     DB      'Ø'+MSIZE/1Ø             ;CP/M MEMORY SIZE
D33A 36                     DB      'Ø'+(MSIZE MOD 1Ø)
D33B 4B2Ø435Ø2F             DB      'K CP/M Vers. '          ;CP/M VERSION NUMBER
D348 32                     DB      CPMREV/1Ø+'Ø'
```

193 LST= UL1

65 LST= CRT

01

41

1100

0100

-2

```
 D349 2E                 DB      '.'
 D34A 32                 DB      (CPMREV MOD 10)+'0'
 D34B 2C20436269         DB      ', Cbios rev '
 D357 332E               DB      REVNUM/10+'0','.'        ;CBIOS REVISION NUMBER
 D359 30                 DB      REVNUM MOD 10+'0'
 D35A 0D0A               DB      ACR,ALF
 D35C 466F722054         DB      'For Thinker Toys Disk Jockey 2D Controller '
 D387 402030             DB      '@ 0'

                         IF      ORIGIN/4096 > 10         ;CONTROLLER ORIGIN (HEX)
 D38A 45                 DB      ORIGIN/4096+'A'-10
                         ELSE
                         DB      ORIGIN/4096+'0'
                         ENDIF

                         IF      (ORIGIN/256 AND 0FH) > 10
                         DB      (ORIGIN/256 AND 0FH)+'A'-10
                         ELSE
 D38B 30                 DB      (ORIGIN/256 AND 0FH)+'0'
                         ENDIF
 D38C 3030482E           DB      '00H.'
 D390 0D0A00             DB      ACR,ALF,0

                ****************************************************************
                *                                                              *
                * UTILITY ROUTINE TO OUTPUT THE MESSAGE POINTED AT BY H&L,     *
                * TERMINATED WITH A NULL.                                      *
                *                                                              *
                ****************************************************************

 D393 7E        MESSAGE MOV     A,M                ;GET A CHARACTER OF THE MESSAGE
 D394 23                INX     H                  ;BUMP TEXT POINTER
 D395 A7                ANA     A                  ;TEST FOR END
 D396 C8                RZ                         ;RETURN IF DONE
 D397 E5                PUSH    H                  ;SAVE POINTER TO TEXT
 D398 4F                MOV     C,A                ;OUTPUT CHARACTER IN C
 D399 CD0CD3            CALL    COUT               ;OUTPUT THE CHARACTER
 D39C E1                POP     H                  ;RESTORE THE POINTER
 D39D C393D3            JMP     MESSAGE            ;CONTINUE UNTIL NULL REACHED

                ****************************************************************
                *                                                              *
                * CBOOT IS THE COLD BOOT LOADER. ALL OF CP/M HAS BEEN LOADED IN *
                * WHEN CONTROL IS PASSED HERE.                                 *
                *                                                              *
                ****************************************************************

 D3A0 310001    CBOOT   LXI     SP,TPA             ;SET UP STACK
 D3A3 CD3AD7            CALL    TINIT              ;INITIALIZE THE TERMINAL
 D3A6 2136D3            LXI     H,PROMPT           ;PREP FOR SENDING SIGNON MESSAGE
 D3A9 CD93D3            CALL    MESSAGE            ;SEND THE PROMPT
 D3AC AF                XRA     A                  ;SELECT DISK A
 D3AD 32D7D8            STA     CPMDRV
 D3B0 320400            STA     CDISK

                ****************************************************************
```

```
                   *                                                                  *
                   * GOCPM IS THE ENTRY POINT FROM COLD BOOTS, AND WARM BOOTS. IT    *
                   * INITIALIZES SOME OF THE LOCATIONS IN PAGE Ø, AND SETS UP THE    *
                   * INITIAL DMA ADDRESS (8ØH).                                       *
                   *                                                                  *
                   ********************************************************************

D3B3 218ØØØ        GOCPM   LXI     H,BUFF             ;SET UP INITIAL DMA ADDRESS
D3B6 CD8AD4                CALL    SETDMA
D3B9 3EC3                  MVI     A,(JMP)            ;INITIALIZE JUMP TO WARM BOOT
D3BB 32ØØØØ                STA     WBOT
D3BE 32Ø5ØØ                STA     ENTRY              ;INITIALIZE JUMP TO BDOS
D3C1 21Ø3D3                LXI     H,WBOOTE           ;ADDRESS IN WARM BOOT JUMP
D3C4 22Ø1ØØ                SHLD    WBOT+1
D3C7 21Ø6C5                LXI     H,BDOS+6           ;ADDRESS IN BDOS JUMP
D3CA 22Ø6ØØ                SHLD    ENTRY+1
D3CD AF                    XRA     A                  ;A <- Ø
D3CE 32DCD8                STA     BUFSEC             ;DISK JOCKEY BUFFER EMPTY
D3D1 32D5D5                STA     BUFWRTN            ;SET BUFFER NOT DIRTY FLAG
D3D4 3AØ4ØØ                LDA     CDISK              ;JUMP TO CP/M WITH CURRENTLY SELECTED DISK IN C
D3D7 4F                    MOV     C,A
D3D8 11FBD3                LXI     D,CMNDBEG          ;BEGINNING OF INITIAL COMMAND
D3DB 21Ø8BD                LXI     H,CCP+8            ;COMMAND BUFFER
D3DE 3EØ1                  MVI     A,CMNDEND-CMNDBEG+1 ;LENGTH OF COMMAND
D3EØ 32Ø7BD                STA     CCP+7
D3E3 47                    MOV     B,A
D3E4 CD37D6                CALL    MOVLOP
D3E7 3AF9D3                LDA     CWFLG
D3EA A7                    ANA     A
D3EB 3AFAD3                LDA     AUTOFLG
D3EE CAF2D3                JZ      CLDBOT
D3F1 1F                    RAR
D3F2 1F            CLDBOT  RAR
D3F3 DAØØBD                JC      CCP
D3F6 C3Ø3BD                JMP     CCP+3              ;ENTER CP/M

D3F9 ØØ            CWFLG   DB      Ø                  ;COLD/WARM BOOT FLAG

                   ********************************************************************
                   *                                                                  *
                   * THE FOLLOWING BYTE DETERMINES IF AN INITIAL COMMAND IS TO BE    *
                   * GIVEN TO CP/M ON WARM OR COLD BOOTS. THE VALUE OF THE BYTE IS   *
                   * USED TO GIVE THE COMMAND TO CP/M:                                *
                   *                                                                  *
                   * Ø = NEVER GIVE COMMAND.                                          *
                   * 1 = GIVE COMMAND ON COLD BOOTS ONLY.                             *
                   * 2 = GIVE THE COMMAND ON WARM BOOTS ONLY.                         *
                   * 3 = GIVE THE COMMAND ON WARM AND COLD BOOTS.                     *
                   *                                                                  *
                   ********************************************************************

D3FA Ø1            AUTOFLG DB      1                  ;AUTO COMMAND FEATURE

                   ********************************************************************
                   *                                                                  *
                   * IF THERE IS A COMMAND INSERTED HERE, IT WILL BE GIVEN IF THE    *
                   * AUTO FEATURE IS ENABLED.                                *
                   *       FOR EXAMPLE:                                      *
                   *                                                         *
                   *       CMNDBEG DB      'MBASIC MYPROG'                   *
                   *       CMNDEND DB      Ø                                 *
                   *                                                         *
                   * WILL EXECUTE MICROSOFT BASIC, AND MBASIC WILL EXECUTE THE  *
                   * "MYPROG" BASIC PROGRAM.                                 *
                   *                                                         *
                   ***********************************************************

                   CMNDBEG DB      ''               ;INITIAL COMMAND GOES HERE
 D3FB ØØ           CMNDEND DB      Ø

                   ***********************************************************
                   *                                                         *
                   * WBOOT LOADS IN ALL OF CP/M EXCEPT THE CBIOS, THEN INITIALIZES *
                   * SYSTEM PARAMETERS AS IN COLD BOOT. SEE THE COLD BOOT LOADER  *
                   * LISTING FOR EXACTLY WHAT HAPPENS DURING WARM AND COLD BOOTS. *
                   *                                                         *
                   ***********************************************************

 D3FC 31ØØØ1       WBOOT   LXI     SP,TPA           ;SET UP STACK POINTER
 D3FF 3EØ1                 MVI     A,1
 D4ØØ =            WFLG    EQU     $-1              ;TEST IF BEGINNING OR
 D4Ø1 A7                   ANA     A                ;        ENDING A WARM BOOT
 D4Ø2 3EØ1                 MVI     A,1
 D4Ø4 32ØØD4               STA     WFLG
 D4Ø7 32F9D3               STA     CWFLG            ;SET COLD/WARM BOOT FLAG
 D4ØA CAB3D3               JZ      GOCPM
 D4ØD AF                   XRA     A
 D4ØE 32ØØD4               STA     WFLG
 D411 4F                   MOV     C,A
 D412 CD33D3               CALL    DJDRV            ;SELECT DRIVE A
 D415 ØEØØ                 MVI     C,Ø              ;SELECT SINGLE DENSITY
 D417 CD2DE4               CALL    DJDEN
 D41A ØEØØ                 MVI     C,Ø              ;SELECT SIDE Ø
 D41C CD3ØE4               CALL    DJSIDE
 D41F 3EØF                 MVI     A,15             ;INITIALIZE THE SECTOR TO READ
 D421 323FD4               STA     NEWSEC
 D424 21ØØBC               LXI     H,CCP-1ØØH       ;AND THE DMA ADDRESS
 D427 225ED4               SHLD    NEWDMA
 D42A CD3ED4               CALL    WARMLOD          ;READ IN CP/M
 D42D Ø1ØØC2               LXI     B,CCP+5ØØH       ;LOAD ADDRESS FOR REST OF WARM BOOT
 D43Ø CD12E4               CALL    DJDMA
 D433 ØEØ8                 MVI     C,8
 D435 CDØFE4               CALL    DJSEC
 D438 CD72D4               CALL    WARMRD
 D43B C3Ø3C2               JMP     CCP+5Ø3H

 D43E 3EØF         WARMLOD MVI     A,15             ;PREVIOUS SECTOR
 D43F =            NEWSEC  EQU     $-1
 D44Ø 3C                   INR     A                ;UPDATE THE PREVIOUS SECTOR
 D441 3C                   INR     A
 D442 FE1B                 CPI     27               ;WAS IT THE LAST ?
 D444 DA56D4               JC      NOWRAP
 D447 D609                SUI     9                 ;YES
 D449 FE13                CPI     19
 D44B C8                  RZ
 D44C 2A5ED4              LHLD    NEWDMA
 D44F 1180FB              LXI     D,-480H
 D452 19                  DAD     D
 D453 225ED4              SHLD    NEWDMA
 D456 323FD4      NOWRAP  STA     NEWSEC            ;SAVE THE NEW SECTOR TO READ
 D459 4F                  MOV     C,A
 D45A CD0FE4              CALL    DJSEC
 D45D 2100BC              LXI     H,CCP-100H        ;GET THE PREVIOUS DMA ADDRESS
 D45E =           NEWDMA  EQU     $-2
 D460 110001              LXI     D,100H            ;UPDATE THE DMA ADDRESS
 D463 19                  DAD     D
 D464 225ED4              SHLD    NEWDMA            ;SAVE THE DMA ADDRESS
 D467 44                  MOV     B,H
 D468 4D                  MOV     C,L
 D469 CD12E4              CALL    DJDMA             ;SET THE DMA ADDRESS
 D46C CD72D4              CALL    WARMRD
 D46F C33ED4              JMP     WARMLOD

 D472 01000A      WARMRD  LXI     B,RETRIES*100H+0;MAXIMUM # OF ERRORS
 D475 C5          WRMREAD PUSH    B
 D476 CD0CE4              CALL    DJTRK             ;SET THE TRACK
 D479 CD15E4              CALL    DJREAD            ;READ THE SECTOR
 D47C C1                  POP     B
 D47D D0                  RNC                       ;CONTINUE IF SUCCESSFUL
 D47E 05                  DCR     B
 D47F C275D4              JNZ     WRMREAD           ;KEEP TRYING
 D482 C32AE4              JMP     DJERR

                  *****************************************************************
                  *                                                               *
                  * SETSEC JUST SAVES THE DESIRED SECTOR TO SEEK TO UNTIL AN      *
                  * ACTUAL READ OR WRITE IS ATTEMPTED.                            *
                  *                                                               *
                  *****************************************************************

 D485 79          SETSEC  MOV     A,C               ;SAVE THE SECTOR NUMBER
 D486 32D6D8              STA     CPMSEC            ;CP/M SECTOR #
 D489 C9                  RET

                  *****************************************************************
                  *                                                               *
                  * SETDMA SAVES THE DMA ADDRESS FOR THE DATA TRANSFER.           *
                  *                                                               *
                  *****************************************************************

 D48A 60          SETDMA  MOV     H,B               ;HL <- BC
 D48B 69                  MOV     L,C
 D48C 22B5D5              SHLD    CPMDMA            ;CP/M DMA ADDRESS
 D48F C9                  RET

                  *****************************************************************
                  *                                                               *
                  * HOME IS TRANSLATED INTO A SEEK TO TRACK ZERO.                 *
```

```
                    *                                                                *
                    ******************************************************************

D49Ø ØEØØ           HOME     MVI     C,Ø                ;TRACK TO SEEK TO

                    ******************************************************************
                    *                                                                *
                    * SETTRK SAVES THE TRACK # TO SEEK TO. NOTHING IS DONE AT THIS  *
                    * POINT, EVERYTHING IS DEFFERED UNTIL A READ OR WRITE.          *
                    *                                                                *
                    ******************************************************************

D492 79             SETTRK   MOV     A,C                ;A <- TRACK #
D493 32D8D8                  STA     CPMTRK             ;CP/M TRACK #
D496 C9                      RET

                    ******************************************************************
                    *                                                                *
                    * SECTRAN TRANSLATES A LOGICAL SECTOR # INTO A PHYSICAL SECTOR  *
                    * #.                                                             *
                    *                                                                *
                    ******************************************************************

D497 Ø3             SECTRAN INX      B
D498 D5                      PUSH    D                  ;SAVE TABLE ADDRESS
D499 C5                      PUSH    B                  ;SAVE SECTOR #
D49A CD41D5                  CALL    GETDPB             ;GET DPB ADDRESS INTO HL
D49D 7E                      MOV     A,M                ;GET # OF CP/M SECTORS/TRACK
D49E B7                      ORA     A                  ;CLEAR CARY
D49F 1F                      RAR                        ;DIVIDE BY TWO
D4AØ 91                      SUB     C
D4A1 F5                      PUSH    PSW                ;SAVE ADJUSTED SECTOR
D4A2 FAAED4                  JM      SIDETWO
D4A5 F1             SIDEA    POP     PSW                ;DISCARD ADJUSTED SECTOR
D4A6 C1                      POP     B                  ;RESTORE SECTOR REQUESTED
D4A7 D1                      POP     D                  ;RESTOR ADDRESS OF XLT TABLE
D4A8 EB             SIDEONE XCHG                        ;HL <- &(TRANSLATION TABLE)
D4A9 Ø9                      DAD     B                  ;BC = OFFSET INTO TABLE
D4AA 6E                      MOV     L,M                ;HL <- PHYSICAL SECTOR
D4AB 26ØØ                    MVI     H,Ø
D4AD C9                      RET

D4AE Ø1ØFØØ         SIDETWO LXI      B,15               ;OFFSET TO SIDE BIT
D4B1 Ø9                      DAD     B
D4B2 7E                      MOV     A,M
D4B3 E6Ø8                    ANI     8                  ;TEST FOR DOUBLE SIDED
D4B5 CAA5D4                  JZ      SIDEA              ;MEDIA IS ONLY SINGLE SIDED
D4B8 F1                      POP     PSW                ;RETRIEVE ADJUSTED SECTOR
D4B9 C1                      POP     B
D4BA 2F                      CMA                        ;MAKE SECTOR REQUEST POSITIVE
D4BB 3C                      INR     A
D4BC 4F                      MOV     C,A                ;MAKE NEW SECTOR THE REQUESTED SECTOR
D4BD D1                      POP     D
D4BE CDA8D4                  CALL    SIDEONE
D4C1 3E8Ø                    MVI     A,8ØH              ;SIDE TWO BIT
D4C3 B5                      ORA     L                  ;       AND SECTOR
```

```
 D4C4 6F                 MOV     L,A
 D4C5 C9                 RET

                 *****************************************************************
                 *                                                               *
                 * SETDRV SELECTS THE NEXT DRIVE TO BE USED IN READ/WRITE        *
                 * OPERATIONS. IF THE DRIVE HAS NEVER BEEN SELECTED BEFORE, A    *
                 * PARAMETER TABLE IS CREATED WHICH CORRECTLY DESCRIBES THE      *
                 * DISKETTE CURRENTLY IN THE DRIVE. DISKETTES CAN BE OF FOUR     *
                 * DIFFERENT SECTOR SIZES:                                       *
                 *       1) 128 BYTES SINGLE DENSITY.                            *
                 *       2) 256 BYTES DOUBLE DENSITY.                            *
                 *       3) 512 BYTES DOUBLE DENSITY.                            *
                 *       4) 1Ø24 BYTES DOUBLE DENSITY.                           *
                 *                                                               *
                 *****************************************************************

 D4C6 79         SETDRV  MOV     A,C             ;SAVE THE DRIVE #
 D4C7 32D7D8             STA     CPMDRV
 D4CA FEØ4               CPI     MAXDISK         ;CHECK FOR A VALID DRIVE #
 D4CC D23DD5             JNC     ZRET            ;ILLEGAL DRIVE #
 D4CF 7B                 MOV     A,E             ;TEST IF DRIVE EVER LOGGED IN BEFORE
 D4DØ E6Ø1               ANI     1
 D4D2 C224D5             JNZ     SETDRV1         ;BIT Ø OF E = Ø -> NEVER SELECTED BEFORE
 D4D5 3EØ1               MVI     A,1             ;SELECT SECTOR 1 OF TRACK 1
 D4D7 32D9D8             STA     TRUESEC
 D4DA 32D8D8             STA     CPMTRK
 D4DD CD2ØD6             CALL    FILL            ;FLUSH BUFFER AND REFILL
 D4EØ DA3DD5             JC      ZRET            ;TEST FOR ERROR RETURN
 D4E3 CD27E4             CALL    DJSTAT          ;GET STATUS ON CURRENT DRIVE
 D4E6 E6ØC               ANI     ØCH             ;STRIP OFF UNWANTED BITS
 D4E8 F5                 PUSH    PSW             ;USED TO SELECT A DPB
 D4E9 1F                 RAR
 D4EA 215AD5             LXI     H,XLTS          ;TABLE OF XLT ADDRESSES
 D4ED 5F                 MOV     E,A
 D4EE 16ØØ               MVI     D,Ø
 D4FØ 19                 DAD     D
 D4F1 E5                 PUSH    H               ;SAVE POINTER TO PROPER XLT
 D4F2 CD41D5             CALL    GETDPB          ;GET DPH POINTER INTO DE
 D4F5 EB                 XCHG                    ;
 D4F6 D1                 POP     D
 D4F7 Ø6Ø2               MVI     B,2             ;NUMBER OF BYTES TO MOVE
 D4F9 CD37D6             CALL    MOVLOP          ;MOVE THE ADDRESS OF XLT
 D4FC 11Ø8ØØ             LXI     D,8             ;OFFSET TO DPB POINTER
 D4FF 19                 DAD     D               ;HL <- &DPH.DPB
 D5ØØ E5                 PUSH    H
 D5Ø1 2AØ7EØ             LHLD    ORIGIN+7        ;GET ADDRESS OF DJ TERMINAL OUT ROUTINE
 D5Ø4 23                 INX     H               ;BUMP TO LOOK AT ADDRESS OF
                                                 ;       UART STATUS LOCATION
 D5Ø5 7E                 MOV     A,M
 D5Ø6 EEØ3               XRI     3               ;ADJUST FOR PROPER REV DJ
 D5Ø8 6F                 MOV     L,A
 D5Ø9 26E3               MVI     H,(ORIGIN+3ØØH)/1ØØH
 D5ØB 7E                 MOV     A,M
 D5ØC E6Ø8               ANI     DBLSID          ;CHECK DOUBLE SIDED BIT
 D5ØE 1116D8             LXI     D,DPB128S       ;BASE FOR SINGLE SIDED DPB'S
```

```
D511 C217D5              JNZ     SIDEOK
D514 1156D8              LXI     D,DPB128D           ;BASE OF DOUBLE SIDED DPB'S
D517 EB         SIDEOK   XCHG                        ;HL <- DBP BASE, DE <- &DPH.DPB
D518 D1                  POP     D                   ;RESTORE DE (POINTER INTO DPH)
D519 F1                  POP     PSW                 ;OFFSET TO CORRECT DPB
D51A 17                  RAL
D51B 17                  RAL
D51C 4F                  MOV     C,A
D51D Ø6ØØ                MVI     B,Ø
D51F Ø9                  DAD     B
D52Ø EB                  XCHG                        ;PUT DPB ADDRESS IN DPH
D521 73                  MOV     M,E
D522 23                  INX     H
D523 72                  MOV     M,D
D524 CD41D5     SETDRV1  CALL    GETDPB              ;GET ADDRESS OF DPB IN HL
D527 Ø1ØFØØ              LXI     B,15                ;OFFSET TO SECTOR SIZE
D52A Ø9                  DAD     B
D52B 7E                  MOV     A,M                 ;GET SECTOR SIZE
D52C E6Ø7                ANI     7H
D52E 326ED5              STA     SECSIZ
D531 7E                  MOV     A,M
D532 1F                  RAR
D533 1F                  RAR
D534 1F                  RAR
D535 1F                  RAR
D536 E6ØF                ANI     ØFH
D538 32A4D5              STA     SECPSEC
D53B EB                  XCHG                        ;HL <- DPH
D53C C9                  RET

D53D 21ØØØØ     ZRET     LXI     H,Ø                 ;SELDRV ERROR EXIT
D54Ø C9                  RET

                *****************************************************************
                *                                                               *
                * GETDPB RETURNS HL POINTING TO THE DPB OF THE CURRENTLY        *
                * SELECTED DRIVE, DE POINTING TO DPH.                           *
                *                                                               *
                *****************************************************************

D541 3AD7D8     GETDPB   LDA     CPMDRV              ;GET DRIVE #
D544 6F                  MOV     L,A                 ;FORM OFFSET
D545 26ØØ                MVI     H,Ø
D547 29                  DAD     H
D548 29                  DAD     H
D549 29                  DAD     H
D54A 29                  DAD     H
D54B 1196D8              LXI     D,DPZERO            ;BASE OF DPH'S
D54E 19                  DAD     D
D54F E5                  PUSH    H                   ;SAVE ADDRESS OF DPH
D55Ø 11ØAØØ              LXI     D,1Ø                ;OFFSET TO DPB
D553 19                  DAD     D
D554 7E                  MOV     A,M                 ;GET LOW BYTE OF DPB ADDRESS
D555 23                  INX     H
D556 66                  MOV     H,M                 ;GET LOW BYTE OF DPB
D557 6F                  MOV     L,A
```

```
 D558 D1               POP     D
 D559 C9               RET

                ****************************************************************
                *                                                              *
                * XLTS IS A TABLE OF ADDRESS THAT POINT TO EACH OF THE XLT     *
                * TABLES FOR EACH SECTOR SIZE.                                 *
                *                                                              *
                ****************************************************************

 D55A 48D7      XLTS   DW      XLT128          ;XLT FOR 128 BYTE SECTORS
 D55C 63D7             DW      XLT256          ;XLT FOR 256 BYTE SECTORS
 D55E 98D7             DW      XLT512          ;XLT FOR 512 BYTE SECTORS
 D560 D5D7             DW      XLT124          ;XLT FOR 1024 BYTE SECTORS

                ****************************************************************
                *                                                              *
                * WRITE ROUTINE MOVES DATA FROM MEMORY INTO THE BUFFER. IF THE *
                * DESIRED CP/M SECTOR IS NOT CONTAINED IN THE DISK BUFFER, THE *
                * BUFFER IS FIRST FLUSHED TO THE DISK IF IT HAS EVER BEEN      *
                * WRITTEN INTO, THEN A READ IS PERFORMED INTO THE BUFFER TO GET *
                * THE DESIRED SECTOR. ONCE THE CORRECT SECTOR IS IN MEMORY, THE *
                * BUFFER WRITTEN INDICATOR IS SET, SO THE BUFFER WILL BE       *
                * FLUSHED, THEN THE DATA IS TRANSFERRED INTO THE BUFFER.       *
                *                                                              *
                ****************************************************************

 D562 79        WRITE  MOV     A,C             ;SAVE WRITE COMMAND TYPE
 D563 32CCD5           STA     WRITTYP
 D566 3E01             MVI     A,1             ;SET WRITE COMMAND
 D568 06               DB      (MVI) OR (B*8)  ;THIS "MVI B" INSTRUCTION  CAUSES
                                               ;        THE FOLLOWING "XRA A" TO
                                               ;        BE SKIPPED OVER.

                ****************************************************************
                *                                                              *
                * READ ROUTINE TO BUFFER DATA FROM THE DISK. IF THE SECTOR     *
                * REQUESTED FROM CP/M IS IN THE BUFFER, THEN THE DATA IS SIMPLY *
                * TRANSFERRED FROM THE BUFFER TO THE DESIRED DMA ADDRESS. IF   *
                * THE BUFFER DOES NOT CONTAIN THE DESIRED SECTOR, THE BUFFER IS *
                * FLUSHED TO THE DISK IF IT HAS EVER BEEN WRITTEN INTO, THEN   *
                * FILLED WITH THE SECTOR FROM THE DISK THAT CONTAINS THE       *
                * DESIRED CP/M SECTOR.                                         *
                *                                                              *
                ****************************************************************

 D569 AF        READ   XRA     A               ;SET THE COMMAND TYPE TO READ
 D56A 32B8D5           STA     RDWR            ;SAVE COMMAND TYPE

                ****************************************************************
                *                                                              *
                * REDWRT CALCULATES THE PHYSICAL SECTOR ON THE DISK THAT       *
                * CONTAINS THE DESIRED CP/M SECTOR, THEN CHECKS IF IT IS THE   *
                * SECTOR CURRENTLY IN THE BUFFER. IF NO MATCH IS MADE, THE     *
                * BUFFER IS FLUSHED IF NECESSARY AND THE CORRECT SECTOR READ   *
                * FROM THE DISK.                                               *
```

```
                 *                                                                *
                 ******************************************************************

D56D 0600        REDWRT  MVI     B,0                 ;THE 0 IS MODIFIED TO CONTAIN THE LOG2
D56E =           SECSIZ  EQU     $-1                 ;       OF THE PHYSICAL SECTOR SIZE/128
                                                     ;       ON THE CURRENTLY SELECTED DISK.
D56F 3AD6D8              LDA     CPMSEC              ;GET THE DESIRED CP/M SECTOR #
D572 F5                  PUSH    PSW                 ;TEMPORARY SAVE
D573 E680                ANI     80H                 ;SAVE ONLY THE SIDE BIT
D575 4F                  MOV     C,A                 ;REMEMBER THE SIDE
D576 F1                  POP     PSW                 ;GET THE SECTOR BACK
D577 E67F                ANI     7FH                 ;FORGET THE SIDE BIT
D579 3D                  DCR     A                   ;TEMPORARY ADJUSTMENT
D57A 05          DIVLOOP DCR     B                   ;UPDATE REPEAT COUNT
D57B CA83D5              JZ      DIVDONE
D57E B7                  ORA     A                   ;CLEAR THE CARY FLAG
D57F 1F                  RAR                         ;DIVIDE THE CP/M SECTOR # BY THE SIZE
                                                     ;       OF THE PHYSICAL SECTORS
D580 C37AD5              JMP     DIVLOOP             ;
D583 3C          DIVDONE INR     A
D584 B1                  ORA     C                   ;RESTORE THE SIDE BIT
D585 32D9D8              STA     TRUESEC             ;SAVE THE PHYSICAL SECTOR NUMBER
D588 21D7D8              LXI     H,CPMDRV            ;POINTER TO DESIRED DRIVE,TRACK, AND SECTOR
D58B 11DAD8              LXI     D,BUFDRV            ;POINTER TO BUFFER DRIVE,TRACK, AND SECTOR
D58E 0604                MVI     B,4                 ;COUNT LOOP
D590 05          DTSLOP  DCR     B                   ;TEST IF DONE WITH COMPARE
D591 CA9FD5              JZ      MOVE                ;YES, MATCH. GO MOVE THE DATA
D594 1A                  LDAX    D                   ;GET A BYTE TO COMPARE
D595 BE                  CMP     M                   ;TEST FOR MATCH
D596 23                  INX     H                   ;BUMP POINTERS TO NEXT DATA ITEM
D597 13                  INX     D
D598 CA90D5              JZ      DTSLOP              ;MATCH, CONTINUE TESTING

                 ******************************************************************
                 *                                                                *
                 * DRIVE, TRACK, AND SECTOR DON'T MATCH, FLUSH THE BUFFER IF      *
                 * NECESSARY AND THEN REFILL.                                     *
                 *                                                                *
                 ******************************************************************

D59B CD20D6              CALL    FILL                ;FILL THE BUFFER WITH CORRECT PHYSICAL SECTOR
D59E D8                  RC                          ;NO GOOD, RETURN WITH ERROR INDICATION

                 ******************************************************************
                 *                                                                *
                 * MOVE HAS BEEN MODIFIED TO CAUSE EITHER A TRANSFER INTO OR OUT *
                 * THE BUFFER.                                                    *
                 *                                                                *
                 ******************************************************************

D59F 3AD6D8      MOVE    LDA     CPMSEC              ;GET THE CP/M SECTOR TO TRANSFER
D5A2 3D                  DCR     A                   ;ADJUST TO PROPER SECTOR IN BUFFER
D5A3 E600                ANI     0                   ;STRIP OFF HIGH ORDERED BITS
D5A4 =           SECPSEC EQU     $-1                 ;THE 0 IS MODIFIED TO REPRESENT THE # OF
                                                     ;       CP/M SECTORS PER PHYSICAL SECTORS
D5A5 6F                  MOV     L,A                 ;PUT INTO HL
```

```
D5A6 26ØØ                MVI     H,Ø
D5A8 29                  DAD     H                  ;FORM OFFSET INTO BUFFER
D5A9 29                  DAD     H
D5AA 29                  DAD     H
D5AB 29                  DAD     H
D5AC 29                  DAD     H
D5AD 29                  DAD     H
D5AE 29                  DAD     H
D5AF 11DDD8              LXI     D,BUFFER           ;BEGINNING ADDRESS OF BUFFER
D5B2 19                  DAD     D                  ;FORM BEGINNING ADDRESS OF SECTOR TO TRANSFER
D5B3 EB                  XCHG                       ;DE = ADDRESS IN BUFFER
D5B4 21ØØØØ              LXI     H,Ø                ;GET DMA ADDRESS, THE Ø IS MODIFIED TO
                                                    ;        CONTAIN THE DMA ADDRESS
D5B5 =          CPMDMA   EQU     $-2
D5B7 3EØØ                MVI     A,Ø                ;THE ZERO GETS MODIFIED TO CONTAIN
                                                    ;        A ZERO IF A READ, OR A 1 IF WRITE
D5B8 =          RDWR     EQU     $-1
D5B9 A7                  ANA     A                  ;TEST WHICH KIND OF OPERATION
D5BA C2C2D5              JNZ     INTO               ;TRANSFER DATA INTO THE BUFFER
D5BD CD35D6     OUTOF    CALL    MOVER
D5CØ AF                  XRA     A
D5C1 C9                  RET

D5C2 EB         INTO     XCHG                       ;
D5C3 CD35D6              CALL    MOVER              ;MOVE THE DATA, HL = DESTINATION
                                                    ;        DE = SOURCE
D5C6 3EØ1                MVI     A,1
D5C8 32D5D5              STA     BUFWRTN            ;SET BUFFER WRITTEN INTO FLAG
D5CB 3EØØ                MVI     A,Ø                ;CHECK FOR DIRECTORY WRITE
D5CC =          WRITTYP  EQU     $-1
D5CD 3D                  DCR     A
D5CE 3EØØ                MVI     A,Ø
D5DØ 32CCD5              STA     WRITTYP            ;SET NO DIRECTORY WRITE
D5D3 CØ                  RNZ                        ;NO ERROR EXIT

                ******************************************************************
                *                                                                *
                * FLUSH WRITES THE CONTENTS OF THE BUFFER OUT TO THE DISK IF     *
                * IT HAS EVER BEEN WRITTEN INTO.                                 *
                *                                                                *
                ******************************************************************

D5D4 3EØØ       FLUSH    MVI     A,Ø                ;THE Ø IS MODIFIED TO REFLECT IF
                                                    ;        THE BUFFER HAS BEEN WRITTEN INTO
D5D5 =          BUFWRTN  EQU     $-1
D5D6 A7                  ANA     A                  ;TEST IF WRITTEN INTO
D5D7 C8                  RZ                         ;NOT WRITTEN, ALL DONE
D5D8 2118E4              LXI     H,DJWRITE          ;WRITE OPERATION

                ******************************************************************
                *                                                                *
                * PREP PREPARES TO READ/WRITE THE DISK. RETRIES ARE ATTEMPTED.   *
                * UPON ENTRY, H&L MUST CONTAIN THE READ OR WRITE OPERATION       *
                * ADDRESS.                                                       *
                *                                                                *
                ******************************************************************
```

```
D5DB AF         PREP    XRA     A               ;RESET BUFFER WRITTEN FLAG
D5DC 32D5D5             STA     BUFWRTN
D5DF 2212D6             SHLD    RETRYOP         ;SET UP THE READ/WRITE OPERATION
D5E2 Ø6ØA               MVI     B,RETRIES       ;MAXIMUM NUMBER OF RETRIES TO ATTEMPT
D5E4 C5         RETRYLP PUSH    B               ;SAVE THE RETRY COUNT
D5E5 3ADAD8             LDA     BUFDRV          ;GET DRIVE NUMBER INVOLVED IN THE OPERATION
D5E8 4F                 MOV     C,A
D5E9 CD33D3             CALL    DJDRV           ;SELECT THE DRIVE
D5EC 3ADBD8             LDA     BUFTRK
D5EF A7                 ANA     A               ;TEST FOR TRACK ZERO
D5FØ 4F                 MOV     C,A
D5F1 C5                 PUSH    B
D5F2 CCØ9E4             CZ      DJHOME          ;HOME THE DRIVE IF TRACK Ø
D5F5 C1                 POP     B               ;RESTORE TRACK #
D5F6 CDØCE4             CALL    DJTRK           ;SEEK TO PROPER TRACK
D5F9 3ADCD8             LDA     BUFSEC          ;GET SECTOR INVOLVED IN OPERATION
D5FC F5                 PUSH    PSW             ;SAVE THE SECTOR #
D5FD Ø7                 RLC                     ;BIT Ø OF A EQUALS SIDE #
D5FE E6Ø1               ANI     1               ;STRIP OFF UNNECESSARY BITS
D6ØØ 4F                 MOV     C,A             ;C <- SIDE #
D6Ø1 CD3ØE4             CALL    DJSIDE          ;SELECT THE SIDE
D6Ø4 F1                 POP     PSW             ;A <- SECTOR #
D6Ø5 E67F               ANI     7FH             ;STRIP OFF SIDE BIT
D6Ø7 4F                 MOV     C,A             ;C <- SECTOR #
D6Ø8 CDØFE4             CALL    DJSEC           ;SET THE SECTOR TO TRANSFER
D6ØB Ø1DDD8             LXI     B,BUFFER        ;SET THE DMA ADDRESS
D6ØE CD12E4             CALL    DJDMA
D611 CD15E4             CALL    DJREAD          ;THE READ OPERATION IS MODIFIED TO WRITE
D612 =          RETRYOP EQU     $-2
D614 C1                 POP     B               ;RESTORE THE RETRY COUNTER
D615 3EØØ               MVI     A,Ø             ;NO ERROR EXIT STATUS
D617 DØ                 RNC                     ;RETURN NO ERROR
D618 Ø5                 DCR     B               ;UPDATE THE RETRY COUNTER
D619 37                 STC                     ;ASSUME RETRY COUNT EXPIRED
D61A 3EFF               MVI     A,ØFFH          ;ERROR RETURN
D61C C8                 RZ
D61D C3E4D5             JMP     RETRYLP         ;TRY AGAIN

                ******************************************************************
                *                                                                *
                * FILL FILLS THE BUFFER WITH A NEW SECTOR FROM THE DISK.         *
                *                                                                *
                ******************************************************************

D62Ø CDD4D5     FILL    CALL    FLUSH           ;FLUSH BUFFER FIRST
D623 D8                 RC                      ;CHECK FOR ERROR
D624 11D7D8             LXI     D,CPMDRV        ;UPDATE THE DRIVE, TRACK, AND SECTOR
D627 21DAD8             LXI     H,BUFDRV
D62A Ø6Ø3               MVI     B,3             ;NUMBER OF BYTES TO MOVE
D62C CD37D6             CALL    MOVLOP          ;COPY THE DATA
D62F 2115E4             LXI     H,DJREAD
D632 C3DBD5             JMP     PREP            ;SELECT DRIVE, TRACK, AND SECTOR.
                                                ;       THEN READ THE BUFFER

                ******************************************************************
```

```
                         *                                                               *
                         * MOVER MOVES 128 BYTES OF DATA. SOURCE POINTER IN DE, DEST     *
                         * POINTER IN HL.                                                *
                         *                                                               *
                         *****************************************************************

D635 0680                MOVER   MVI     B,128               ;LENGTH OF TRANSFER
D637 1A                  MOVLOP  LDAX    D                   ;GET A BTE OF SOURCE
D638 77                          MOV     M,A                 ;MOVE IT
D639 13                          INX     D                   ;BUMP POINTERS
D63A 23                          INX     H
D63B 05                          DCR     B                   ;UPDATE COUNTER
D63C C237D6                      JNZ     MOVLOP              ;CONTINUE MOVING UNTIL DONE
D63F C9                          RET

                         *****************************************************************
                         *                                                               *
                         * TERMINAL DRIVER ROUTINES. IOBYTE IS INITIALIZED BY THE COLD   *
                         * BOOT ROUTINE, TO MODIFY, CHANGE THE "INTIOBY" EQUATE. THE     *
                         * I/O ROUTINES THAT FOLLOW ALL WORK EXACTLY THE SAME WAY. USING *
                         * IOBYTE, THEY OBTAIN THE ADDRESS TO JUMP TO IN ORDER TO EXECUTE*
                         * THE DESIRED FUNCTION. THERE IS A TABLE WITH FOUR ENTRIES FOR  *
                         * EACH OF THE POSSIBLE ASSIGNMENTS FOR EACH DEVICE. TO MODIFY   *
                         * THE I/O ROUTINES FOR A DIFFERENT I/O CONFIGURATION, JUST      *
                         * CHANGE THE ENTRIES IN THE TABLES.                             *
                         *                                                               *
                         *****************************************************************

E403 =                   CITTY   EQU     DJCIN               ;INPUT FROM THE DISK JOCKEY 2D
E406 =                   COTTY   EQU     DJCOUT              ;OUTPUT TO THE DISK JOCKEY 2D

                         *****************************************************************
                         *                                                               *
                         * CONST: GET THE STATUS FOR THE CURRENTLY ASSIGNED CONSOLE      *
                         *        DEVICE. THE CONSOLE DEVICE CAN BE GOTTEN FROM IOBYTE,  *
                         *        THEN A JUMP TO THE CORRECT CONSOLE STATUS ROUTINE IS   *
                         *        PERFORMED.                                             *
                         *                                                               *
                         *****************************************************************

D640 21BAD6              CONST   LXI     H,CSTBLE            ;BEGINNING OF JUMP TABLE
D643 C352D6                      JMP     CONIN1              ;SELECT CORRECT JUMP

                         *****************************************************************
                         *                                                               *
                         * CSREADER: IF THE CONSOLE IS ASSIGNED TO THE READER THEN A     *
                         *           JUMP WILL BE MADE HERE, WHERE ANOTHER JUMP WILL     *
                         *           OCCUR TO THE CORRECT READER STATUS.                 *
                         *                                                               *
                         *****************************************************************

D646 21C2D6              CSREADR LXI     H,CSRTBLE           ;BEGINNING OF READER STATUS TABLE
D649 C36FD6                      JMP     READERA

                         *****************************************************************
                         *                                                               *
```

```
                  * CONIN: TAKE THE CORRECT JUMP FOR THE CONSOLE INPUT ROUTINE.   *
                  *        THE JUMP IS BASED ON THE TWO LEAST SIGNIFICANT BITS OF *
                  *        IOBYTE.                                                *
                  *                                                               *
                  *****************************************************************

D64C CDD4D5       CONIN   CALL    FLUSH           ;FLUSH THE DISK BUFFER
D64F 2192D6               LXI     H,CITBLE        ;BEGINNING OF CHARACTER INPUT TABLE

                  *
                  * ENTRY AT CONIN1 WILL DECODE THE TWO LEAST SIGNIFICANT BITS
                  * OF IOBYTE. THIS IS USED BY CONIN,CONOUT, AND CONST.
                  *

D652 3A0300       CONIN1  LDA     IOBYTE
D655 17                   RAL

                  *
                  * ENTRY AT SELDEV WILL FORM AN OFFSET INTO THE TABLE POINTED
                  * TO BY H&L AND THEN PICK UP THE ADDRESS AND JUMP THERE.
                  *

D656 E606         SELDEV  ANI     6H              ;STRIP OFF UNWANTED BITS
D658 1600                 MVI     D,0             ;FORM OFFSET
D65A 5F                   MOV     E,A
D65B 19                   DAD     D               ;ADD OFFSET
D65C 7E                   MOV     A,M             ;PICK UP HIGH BYTE
D65D 23                   INX     H
D65E 66                   MOV     H,M             ;PICK UP LOW BYTE
D65F 6F                   MOV     L,A             ;FORM ADDRESS
D660 E9                   PCHL                    ;GO THERE !

                  *****************************************************************
                  *                                                               *
                  * CONOUT: TAKE THE PROPER BRANCH ADDRESS BASED ON THE TWO LEAST *
                  *         SIGNIFICANT BITS OF IOBYTE.                  0-1      *
                  *                                                               *
                  *****************************************************************

D661 C5           CONOUT  PUSH    B               ;SAVE THE CHARACTER
D662 CDD4D5               CALL    FLUSH           ;FLUSH THE DISK BUFFER
D665 C1                   POP     B               ;RESTORE THE CHARACTER
D666 219AD6               LXI     H,COTBLE        ;BEGINNING OF THE CHARACTER OUT TABLE
D669 C352D6               JMP     CONIN1          ;DO THE DECODE

                  *****************************************************************
                  *                                                               *
                  * READER: SELECT THE CORRECT READER DEVICE FOR INPUT. THE       *
                  *         READER IS SELECTED FROM BITS 2 AND 3 OF IOBYTE.       *
                  *                                                               *
                  *****************************************************************

D66C 21B2D6       READER  LXI     H,RTBLE         ;BEGINNING OF READER INPUT TABLE

                  *
                  * ENTRY AT READERA WILL DECODE BITS 2 & 3 OF IOBYTE, USED
```

```
                    * BY CSREADER.
                    *

D66F 3A0300         READERA LDA     IOBYTE

                    *
                    * ENTRY AT READER1 WILL SHIFT THE BITS INTO POSITION, USED
                    * BY LIST AND PUNCH.
                    *

D672 1F             READR1  RAR
D673 C356D6                 JMP     SELDEV

                    *****************************************************************
                    *                                                               *
                    * PUNCH: SELECT THE CORRECT PUNCH DEVICE. THE SELECTION COMES   *
                    *        FROM BITS 4&5 OF IOBYTE.                               *
                    *                                                               *
                    *****************************************************************

D676 21AAD6         PUNCH   LXI     H,PTBLE         ;BEGINNING OF PUNCH TABLE
D679 3A0300                 LDA     IOBYTE

                    *
                    * ENTRY AT PNCH1 ROTATES BITS A LITTLE MORE IN PREP FOR
                    * SELDEV, USED BY LIST.
                    *

D67C 1F             PNCH1   RAR
D67D 1F                     RAR
D67E C372D6                 JMP     READR1

                    *****************************************************************
                    *                                                               *
                    * LIST: SELECT A LIST DEVICE BASED ON BITS 6&7 OF IOBYTE        *
                    *                                                               *
                    *****************************************************************

D681 21A2D6         LIST    LXI     H,LTBLE         ;BEGINNING OF THE LIST DEVICE ROUTINES
D684 3A0300         LIST1   LDA     IOBYTE
D687 1F                     RAR
D688 1F                     RAR
D689 C37CD6                 JMP     PNCH1

                    *****************************************************************
                    *                                                               *
                    * LISTST: GET THE STATUS OF THE CURRENTLY ASSIGNED LIST DEVICE  *
                    *                                                               *
                    *****************************************************************

D68C 21CAD6         LISTST  LXI     H,LSTBLE        ;BEGINNING OF THE LIST DEVICE STATUS
D68F C384D6                 JMP     LIST1

                    *****************************************************************
                    *                                                               *
                    * IF CUSTOMIZING I/O ROUTINES IS BEING PERFORMED, THE TABLE     *
```

```
CP/M MACRO ASSEM 2.Ø    #Ø18     *** Cbios For CP/M Ver. 2.2 ***

                        * BELOW SHOULD BE MODIFIED TO REFLECT THE CHANGES. ALL I/O     *
                        * DEVICES ARE DECODED OUT OF IOBYTE AND THE JUMP IS TAKEN FROM *
                        * THE FOLLOWING TABLES.                                        *
                        *                                                              *
                        ****************************************************************

                        *
                        * CONSOLE INPUT TABLE
                        *

 D692 ØØD7              CITBLE  DW      CIUC1            ;INPUT FROM USER CONSOLE 1 (CURRENTLY
                                                         ;       SWBD PARALLEL PORT 4)
 D694 15D7                      DW      CICRT            ;INPUT FROM CRT (CURRENTLY SWITCHBOARD
                                                         ;       SERIAL PORT 1)
 D696 6CD6                      DW      READER           ;INPUT FROM READER (DEPENDS ON READER
                                                         ;       SELECTION)
 D698 Ø3E4                      DW      CITTY            ;INPUT FROM TTY (CURRENTLY INPUT FROM
                                                         ;       DISK JOCKEY 2D)

                        *
                        * CONSOLE OUTPUT TABLE
                        *

 D69A D2D6              COTBLE  DW      COCRT            ;OUTPUT TO CRT (MSDV)
                                                         ;
 D69C D2D6                      DW      COCRT            ;OUTPUT TO CRT (MSDV)
                                                         ;
 D69E 81D6                      DW      LIST             ;OUTPUT TO LIST DEVICE (DEPENDS ON
                                                         ;       BITS 6&7 OF IOBYTE)
 D6AØ Ø6E4                      DW      COTTY            ;OUTPUT TO TTY (CURRENTLY OUTPUT TO
                                                         ;       DISK JOCKEY 2D)
                        *
                        * LIST DEVICE TABLE
                        *

 D6A2 Ø6E4              LTBLE   DW      COTTY            ;OUTPUT TO TTY (CURRENTLY ASSIGNED              COPTA
                                                         ;       BY INTIOBY,OUTPUT TO 2D)
 D6A4 D2D6                      DW      ~~COCRT~~ COPTR  ;OUTPUT TO ~~CRT (MSDV)~~ PRINTER
                                                         ;
 D6A6 D6D6                      DW      COLPT            ;OUTPUT TO LINE PRINTER (CURRENTLY
                                                         ;       SWITCHBOARD SERIAL PORT 1)
 D6A8 E1D6                      DW      COUL1            ;OUTPUT TO USER LINE PRINTER 1 (CURRENTLY       INITIOBY=192
                                                         ;       SWITCHBOARD SERIAL PORT 1)
                                                                                                         11000000
                        *
                        * PUNCH DEVICE TABLE
                        *

 D6AA Ø6E4              PTBLE   DW      COTTY            ;OUTPUT TO THE TTY (CURRENTLY ASSIGNED      ~~COPTA    IN 7     GET PTR READY~~
                                                         ;       BY INTIOBY,OUTPUT TO 2D)            ~~         ANI 2    BIT 2 ONLY~~
 D6AC D6D6                      DW      ~~COPTP~~ COPTR  ;OUTPUT TO ~~PAPER TAPE PUNCH (CURRENTLY~~ PRINTER
                                                         ;       ~~SWITCHBOARD SERIAL PORT 1~~)      ~~         JNZ COPTA WAIT TILL OK TO SEND~~
 D6AE D6D6                      DW      COUP1            ;OUTPUT TO USER PUNCH 1 (CURRENTLY          ~~         JMP COTTY OUTPUT THE CHARACTER~~
                                                         ;       SWITCHBOARD SERIAL PORT 1)
 D6BØ D6D6                      DW      COUP2            ;OUTPUT TO USER PUNCH 2 (CURRNTLLY
                                                         ;       SWITCHBOARD SERIAL PORT 1)
```

```
                              *
                              * READER DEVICE INPUT TABLE
                              *

D6B2 Ø3E4                     RTBLE    DW      CITTY           ;INPUT FROM TTY (CURRENTLY ASSIGNED
                                                               ;        BY INTIOBY, INPUT FROM 2D)
D6B4 15D7                              DW      CIPTR           ;INPUT FROM PAPER TAPE READER (CURRENTLY
                                                               ;        SWITCHBOARD SERIAL PORT 1)
D6B6 15D7                              DW      CIUR1           ;INPUT FROM USER READER 1 (CURRENTLY
                                                               ;        SWITCHBOARD SERIAL PORT 1)
D6B8 15D7                              DW      CIUR2           ;INPUT FROM USER READER 2 (CURRENTLY
                                                               ;        SWITCHBOARD SERIAL PORT 1)

                              *
                              * CONSOLE STATUS TABLE
                              *

D6BA ØCD7                     CSTBLE   DW      CSUC1           ;STATUS FROM SWBD PARALLEL PORT 4, AS
                                                               ;        READ FROM ATTN BIT Ø)
D6BC 29D7                              DW      CSCRT           ;STATUS FROM CRT (CURRENTLY SWITCHBOARD
                                                               ;        SERIAL PORT 1)
D6BE 46D6                              DW      CSREADR         ;STATUS FROM READER (DEPENDS ON READER DEVICE )
                                                               ;
D6CØ 21D7                              DW      CSTTY           ;STATUS OF TTY (CURRENTLY STSTUS FROM
                                                               ;        DISK JOCKEY 2D)
                              *
                              * STATUS FROM READER DEVICE
                              *

D6C2 21D7                     CSRTBLE  DW      CSTTY           ;STATUS FROM TTY (CURRENTLY ASSIGNED
                                                               ;        BY INTIOBY, STATUS OF 2D)
D6C4 29D7                              DW      CSPTR           ;STATUS FROM PAPER TAPE READER (CURRENTLY
                                                               ;        SWITCHBOARD SERIAL PORT 1)
D6C6 29D7                              DW      CSUR1           ;STATUS FROM USER READER 1 (CURRENTLY
                                                               ;        SWITCHBOARD SERIAL PORT 1)
D6C8 29D7                              DW      CSUR2           ;STATUS OF USER READER 2 (CURRENTLY
                                                               ;        SWITCHBOARD SERIAL PORT 1)

                              *
                              * STATUS FROM LIST DEVICE
                              *

D6CA 37D7                     LSTBLE   DW      READY           ;CONSOLE ALWAYS READY
D6CC 37D7                              DW      READY           ;GET LIST STATUS
D6CE 32D7                              DW      LSLPT
D6DØ 32D7                              DW      LSLPT

                              ****************************************************************
                              *                                                              *
                              * ROUTINES FOR MY SYSTEM.   J. J. O'BRIEN                      *
                              *                                                              *
                              ****************************************************************

                              *
                              * MSDV VIDEO DRIVER
```

COP

```
COPTR    IN   2
         ANI  8
         JZ   COPTR
         MOV  A,C
         OUT  0
         RET

COPTR1   IN 5
         ANI 1
         JZ COPTR1
```

~~PTRST IN 5~~
~~ANI 1~~
~~RZ~~
~~JMP READY~~

```
                    *

D6D2 79             COCRT   MOV     A,C                 ;MSDV WANTS DATA IN A
D6D3 C300E8                 JMP     MSDV                ;GO THERE

                    *****************************************************************
                    *                                                               *
                    * THE FOLLOWING EQUATES SET OUTPUT DEVICE TO OUTPUT TO THE      *
                    * SWITCHBOARD SERIAL PORT 1.                                    *
                    *                                                               *
                    *****************************************************************

D6D6 =              COPTP   EQU     $                   ;OUTPUT FROM PAPER TAPE PUNCH
D6D6 =              COUP1   EQU     $                   ;OUTPUT FROM USER PUNCH 1
D6D6 =              COUP2   EQU     $                   ;OUTPUT FROM USER PUNCH 2
D6D6 DB02           COLPT   IN      2                   ;OUTPUT FROM LINE PRINTER,GET STATUS
D6D8 E680                   ANI     80H                 ;WAIT UNTIL OK TO SEND
D6DA CAD6D6                 JZ      COLPT
D6DD 79                     MOV     A,C                 ;OUTPUT THE CHARACTER
D6DE D301                   OUT     1
D6E0 C9                     RET

                    *****************************************************************
                    *                                                               *
                    * CUSTOM I/O PRINTER DRIVER FOR DIABLO PRINTER WITH 1200 BAUD   *
                    * ETX/ACK HANDSHAKE.                                            *
                    *                                                               *
                    *****************************************************************

D6E1 CDD6D6         COUL1   CALL    COLPT               ;OUTPUT THE CHARACTER
D6E4 3AFFD6                 LDA     COUNT
D6E7 3D                     DCR     A
D6E8 32FFD6                 STA     COUNT
D6EB C0                     RNZ
D6EC 3E4E                   MVI     A,78
D6EE 32FFD6                 STA     COUNT
D6F1 0E03                   MVI     C,AETX
D6F3 CDD6D6                 CALL    COLPT
D6F6 CD15D7         PWAIT   CALL    CIPTR
D6F9 FE06                   CPI     AACK
D6FB C2F6D6                 JNZ     PWAIT
D6FE C9                     RET

D6FF 32             COUNT   DB      50

                    ****************************************************************
                    *                                                              *
                    * THE FOLLOWING EQUATES SET THE INPUT TO COME FROM THE SWBD    *
                    * PARALLEL PORT 4, WITH STATUS ON ATTENTION PORT BIT 0.        *
                    *                                                              *
                    ****************************************************************

D700 DB03           CIUC1   IN      3                   ;GET ATTENTION BYTE
D702 E601                   ANI     1                   ;GET BIT 0 ONLY
D704 CA00D7                 JZ      CIUC1               ;WAIT FOR CHARACTER
```

```
D707 DB04               IN      4               ;GET CHARACTER
D709 E67F               ANI     7FH             ;STRIP OFF THE PARITY
D70B C9                 RET

D70C DB03       CSUC1   IN      3               ;GET ATTENTION BYTE
D70E E601               ANI     1               ;GET BIT 0 ONLY
D710 EE01               XRI     1               ;CHANGE POLARITY
D712 C324D7             JMP     STAT            ;RETURN PROPER INDICATION

                ***************************************************************
                *                                                             *
                * THE FOLLOWING EQUATES SET THE INPUT FROM THE DEVICES TO COME *
                * FROM THE SWITCHBOARD SERIAL PORT 1.                         *
                *                                                             *
                ***************************************************************

D715 =          CICRT   EQU     $               ;INPUT FROM CRT
D715 =          CIUR1   EQU     $               ;INPUT FROM USER READER 1
D715 =          CIUR2   EQU     $               ;INPUT FROM USER READER 2
D715 DB02       CIPTR   IN      2               ;INPUT FROM PAPER TAPE READER, GET STATUS
D717 E640               ANI     40H             ;WAIT FOR CHARACTER
D719 CA15D7             JZ      CIPTR
D71C DB01               IN      1
D71E E67F               ANI     7FH             ;STRIP OFF THE PARITY
D720 C9                 RET

                ***************************************************************
                *                                                             *
                * CONSOLE STATUS ROUTINES, TEST IF A CHARACTER HAS ARRIVED.   *
                *                                                             *
                ***************************************************************

D721 CD21E4     CSTTY   CALL    DJTSTAT         ;STATUS FROM DISK JOCKEY 2D
D724 3E00       STAT    MVI     A,0             ;PREP FOR ZERO RETURN
D726 C0                 RNZ                     ;NOTHING FOUND
D727 3D                 DCR     A               ;RETURN WITH 0FFH
D728 C9                 RET

                ***************************************************************
                *                                                             *
                * THE FOLLOWING EQUATES CAUSE THE DEVICES TO GET STATUS FROM  *
                * THE SWITCHBOARD SERIAL PORT 1.                              *
                *                                                             *
                ***************************************************************

D729 =          CSUR1   EQU     $               ;STATUS OF USER READER 1
D729 =          CSUR2   EQU     $               ;STATUS OF USER READER 2
D729 =          CSPTR   EQU     $               ;STATUS OF PAPER TAPE READER
D729 DB02       CSCRT   IN      2               ;STATUS FROM CRT, GET STATUS
D72B E640               ANI     40H             ;STRIP OF DATA READY BIT
D72D EE40               XRI     40H             ;MAKE CORRECT POLARITY
D72F C324D7             JMP     STAT            ;RETURN PROPER INDICATION

                ***************************************************************
                *                                                             *
                * LIST DEVICE STATUS ROUTINES.                                *
```

INPUT FROM DIABLO

190

```
                    *                                                              *
                    ****************************************************************

D732 DBØ2           LSLPT   IN      2                   ;ALL OTHER DEVICES WAIT
D734 E68Ø                   ANI     8ØH
D736 C8                     RZ
D737 3EFF           READY   MVI     A,ØFFH
D739 C9                     RET

                    ****************************************************************
                    *                                                              *
                    * THIS INITIALLIZING ROUTINE SAMPLES BIT Ø OF SWBD PORT 7 TO   *
                    * DETERMINE IF THE KEYBOARD IS PLUGGED IN.  IF THE KEYBOARD IS *
                    * PLUGGED IN, THE LSB RETURNS A Ø. OTHERWISE, IT IS A 1.       *
                    * THIS 1 IS ADDED TO IOBYTE TO CHANGE THE CONSOLE INPUT FROM   *
                    * THE SWBD PARALLEL PORT 4 (THE KEYBOARD) TO THE SWBD SERIAL   *
                    * PORT THAT RECEIVES RS232 DATA FROM THE RS232 TERMINAL.       *
                    *                                                              *
                    ****************************************************************

D73A ØE19           TINIT   MVI     C,CLEAR             ;INITIALIZE THE TERMINAL ROUTINE
D73C DBØ7                   IN      7                   ;GET KEYBOARD INTERLOCK BYTE
D73E E6Ø1                   ANI     1                   ;GET BIT 1 ONLY
D74Ø C6CØ                   ADI     INTIOBY             ;ADD INTIOBY TO KEYBOARD BIT
D742 32Ø3ØØ                 STA     IOBYTE              ;INITIALIZE IOBYTE
D745 C3ØCD3                 JMP     COUT

                    ****************************************************************
                    *                                                              *
                    * XLT TABLES (SECTOR SKEW TABLES) FOR CP/M 2.Ø. THESE TABLES   *
                    * DEFINE THE SECTOR TRANSLATION THAT OCCURS WHEN MAPPING CP/M  *
                    * SECTORS TO PHYSICAL SECTORS ON THE DISK. THERE IS ONE SKEW   *
                    * TABLE FOR EACH OF THE POSSIBLE SECTOR SIZES. CURRENTLY THE   *
                    * TABLES ARE LOCATED ON TRACK Ø SECTORS 6 AND 8. THEY ARE      *
                    * LOADED INTO MEMORY IN THE CBIOS RAM BY THE COLD BOOT ROUTINE. *
                    *                                                              *
                    ****************************************************************

D748 ØØ             XLT128  DB      Ø
D749 Ø1Ø7ØD1319             DB      1,7,13,19,25
D74E Ø5ØB1117               DB      5,11,17,23
D752 Ø3Ø9ØF15               DB      3,9,15,21
D756 Ø2Ø8ØE141A             DB      2,8,14,2Ø,26
D75B Ø6ØC1218               DB      6,12,18,24
D75F Ø4ØA1Ø16               DB      4,1Ø,16,22

D763 ØØ             XLT256  DB      Ø
D764 Ø1Ø2131425             DB      1,2,19,2Ø,37,38
D76A Ø3Ø4151627             DB      3,4,21,22,39,4Ø
D77Ø Ø5Ø6171829             DB      5,6,23,24,41,42
D776 Ø7Ø8191A2B             DB      7,8,25,26,43,44
D77C Ø9ØA1B1C2D             DB      9,1Ø,27,28,45,46
D782 ØBØC1D1E2F             DB      11,12,29,3Ø,47,48
D788 ØDØE1F2Ø31             DB      13,14,31,32,49,5Ø
D78E ØF1Ø212233             DB      15,16,33,34,51,52
D794 11122324               DB      17,18,35,36
```

```
D798 00         XLT512  DB      0
D799 0102030411         DB      1,2,3,4,17,18,19,20
D7A1 2122232431         DB      33,34,35,36,49,50,51,52
D7A9 0506070815         DB      5,6,7,8,21,22,23,24
D7B1 2526272835         DB      37,38,39,40,53,54,55,56
D7B9 090A0B0C19         DB      9,10,11,12,25,26,27,28
D7C1 292A2B2C39         DB      41,42,43,44,57,58,59,60
D7C9 0D0E0F101D         DB      13,14,15,16,29,30,31,32
D7D1 2D2E2F30           DB      45,46,47,48

D7D5 00         XLT124  DB      0
D7D6 0102030405         DB      1,2,3,4,5,6,7,8
D7DE 191A1B1C1D         DB      25,26,27,28,29,30,31,32
D7E6 3132333435         DB      49,50,51,52,53,54,55,56
D7EE 090A0B0C0D         DB      9,10,11,12,13,14,15,16
D7F6 2122232425         DB      33,34,35,36,37,38,39,40
D7FE 393A3B3C3D         DB      57,58,59,60,61,62,63,64
D806 1112131415         DB      17,18,19,20,21,22,23,24
D80E 292A2B2C2D         DB      41,42,43,44,45,46,47,48

                *****************************************************************
                *                                                               *
                * EACH OF THE FOLLOWING TABLES DESCRIBES A DISKETTE WITH THE    *
                * SPECIFIED CHARACTERISTICS. THE TABLES ARE CURRENTLY STORED    *
                * ON TRACK 0 SECTOR 13. THEY ARE READ INTO MEMORY BY THE GOCPM  *
                * ROUTINE IN THE CBIOS FOR CP/M VER 2.0.                        *
                *                                                               *
                *****************************************************************


                *****************************************************************
                *                                                               *
                * THE FOLLOWING DPB DEFINES A  DISKETTE FOR 128 BYTE SECTORS,   *
                * SINGLE DENSITY, AND SINGLE SIDED.                             *
                *                                                               *
                *****************************************************************

D816 1A00       DPB128S DW      26                ;CP/M SECTORS/TRACK
D818 03                 DB      3                 ;BSH
D819 07                 DB      7                 ;BLM
D81A 00                 DB      0                 ;EXM
D81B F200               DW      242               ;DSM
D81D 3F00               DW      63                ;DRM
D81F C0                 DB      0C0H              ;AL0
D820 00                 DB      0                 ;AL1
D821 1000               DW      16                ;CKS
D823 0200               DW      2                 ;OFF
D825 01                 DB      1H                ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                  ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                  ;8 IF DOUBLE SIDED.

                *****************************************************************
                *                                                               *
                * THE FOLLOWING DPB DEFINES A DISKETTE FOR 256 BYTE SECTORS,    *
                * DOUBLE DENSITY, AND SINGLE SIDED.                             *
```

```
                         *                                                              *
                         ****************************************************************

D826 3400                DPB256S DW      52                          ;CP/M SECTORS/TRACK
D828 04                          DB      4                           ;BSH
D829 0F                          DB      15                          ;BLM
D82A 00                          DB      0                           ;EXM
D82B F200                        DW      242                         ;DSM
D82D 7F00                        DW      127                         ;DRM
D82F C0                          DB      0C0H                        ;AL0
D830 00                          DB      0                           ;AL1
D831 2000                        DW      32                          ;CKS
D833 0200                        DW      2                           ;OFF
D835 12                          DB      12H                         ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                                     ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                                     ;8 IF DOUBLE SIDED.

                         ****************************************************************
                         *                                                              *
                         * THE FOLLOWING DPB DEFINES A DISKETTE AS 512 BYTE SECTORS,    *
                         * DOUBLE DENSITY, AND SINGLE SIDED.                            *
                         *                                                              *
                         ****************************************************************

D836 3C00                DPB512S DW      60                          ;CP/M SECTORS/TRACK
D838 04                          DB      4                           ;BSH
D839 0F                          DB      15                          ;BLM
D83A 00                          DB      0                           ;EXM
D83B 1801                        DW      280                         ;DSM
D83D 7F00                        DW      127                         ;DRM
D83F C0                          DB      0C0H                        ;AL0
D840 00                          DB      0                           ;AL1
D841 2000                        DW      32                          ;CKS
D843 0200                        DW      2                           ;OFF
D845 33                          DB      33H                         ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                                     ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                                     ;8 IF DOUBLE SIDED.

                         ****************************************************************
                         *                                                              *
                         * THE FOLLOWING DPB DEFINES A DISKETTE AS 1024 BYTE SECTORS,   *
                         * DOUBLE DENSITY, AND SINGLE SIDED.                            *
                         *                                                              *
                         ****************************************************************

D846 4000                DP1024S DW      64                          ;CP/M SECTORS/TRACK
D848 04                          DB      4                           ;BSH
D849 0F                          DB      15                          ;BLM
D84A 00                          DB      0                           ;EXM
D84B 2B01                        DW      299                         ;DSM
D84D 7F00                        DW      127                         ;DRM
D84F C0                          DB      0C0H                        ;AL0
D850 00                          DB      0                           ;AL1
D851 2000                        DW      32                          ;CKS
D853 0200                        DW      2                           ;OFF
D855 74                          DB      74H                         ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                    ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                    ;8 IF DOUBLE SIDED.

                       ****************************************************************
                       *                                                              *
                       * THE FOLLOWING DPB DEFINES A DISKETTE FOR 128 BYTE SECTORS,   *
                       * SINGLE DENSITY, AND DOUBLE SIDED.                            *
                       *                                                              *
                       ****************************************************************

D856 34ØØ              DPB128D DW      52              ;CP/M SECTORS/TRACK
D858 Ø4                        DB      4               ;BSH
D859 ØF                        DB      15              ;BLM
D85A Ø1                        DB      1               ;EXM
D85B F2ØØ                      DW      242             ;DSM
D85D 7FØØ                      DW      127             ;DRM
D85F CØ                        DB      ØCØH            ;ALØ
D86Ø ØØ                        DB      Ø               ;AL1
D861 2ØØØ                      DW      32              ;CKS
D863 Ø2ØØ                      DW      2               ;OFF
D865 Ø9                        DB      9H

                       ****************************************************************
                       *                                                              *
                       * THE FOLLOWING DPB DEFINES A DISKETTE AS 256 BYTE SECTORS,    *
                       * DOUBLE DENSITY, AND DOUBLE SIDED.                            *
                       *                                                              *
                       ****************************************************************

D866 68ØØ              DPB256D DW      1Ø4             ;CP/M SECTORS/TRACK
D868 Ø4                        DB      4               ;BSH
D869 ØF                        DB      15              ;BLM
D86A ØØ                        DB      Ø               ;EXM
D86B E6Ø1                      DW      486             ;DSM
D86D FFØØ                      DW      255             ;DRM
D86F FØ                        DB      ØFØH            ;ALØ
D87Ø ØØ                        DB      Ø               ;AL1
D871 4ØØØ                      DW      64              ;CKS
D873 Ø2ØØ                      DW      2               ;OFF
D875 1A                        DB      1AH

                       ****************************************************************
                       *                                                              *
                       * THE FOLLOWING DPB DEFINES A DISKETTE AS 512 BYTE SECTORS,    *
                       * DOUBLE DENSITY, AND DOUBLE SIDED.                            *
                       *                                                              *
                       ****************************************************************

D876 78ØØ              DPB512D DW      12Ø             ;CP/M SECTORS/TRACK
D878 Ø4                        DB      4               ;BSH
D879 ØF                        DB      15              ;BLM
D87A ØØ                        DB      Ø               ;EXM
D87B 31Ø2                      DW      561             ;DSM
D87D FFØØ                      DW      255             ;DRM
D87F FØ                        DB      ØFØH            ;ALØ
D88Ø ØØ                        DB      Ø               ;AL1
```

```
 D881 4ØØØ              DW      64              ;CKS
 D883 Ø2ØØ              DW      2               ;OFF
 D885 3B                DB      3BH

                *****************************************************************
                *                                                               *
                * THE FOLLOWING DPB DEFINES A DISKETTE AS 1Ø24 BYTE SECTORS,    *
                * DOUBLE DENSITY, AND DOUBLE SIDED.                             *
                *                                                               *
                *****************************************************************

 D886 8ØØØ      DP1Ø24D DW      128             ;CP/M SECTORS/TRACK
 D888 Ø4                DB      4               ;BSH
 D889 ØF                DB      15              ;BLM
 D88A ØØ                DB      Ø               ;EXM
 D88B 57Ø2              DW      599             ;DSM
 D88D FFØØ              DW      255             ;DRM
 D88F FØ                DB      ØFØH            ;ALØ
 D89Ø ØØ                DB      Ø               ;AL1
 D891 4ØØØ              DW      64              ;CKS
 D893 Ø2ØØ              DW      2               ;OFF
 D895 7C                DB      7CH

                *****************************************************************
                *                                                               *
                * CP/M DISK PARAMETER HEADERS, UNITIALIZED.                     *
                *                                                               *
                *****************************************************************

 D896 ØØØØ      DPZERO  DW      Ø               ;ADDRESS OF TRANSLATION TABLE (FILLED
                                                ;       IN BY SETDRV)
 D898 ØØØØØØØØØØ        DW      Ø,Ø,Ø           ;USED BY BDOS
 D89E Ø9DF              DW      DIRBUF          ;ADDRESS OF DIRECTORY BUFFER
 D8AØ ØØØØ              DW      Ø               ;ADDRESS OF DPB (FILLED IN BY SETDRV)
 D8A2 Ø9DE              DW      CSVØ            ;DIRECTORY CHECK VECTOR
 D8A4 DDDC              DW      ALVØ            ;ALLOCATION VECTOR

 D8A6 ØØØØ      DPONE   DW      Ø
 D8A8 ØØØØØØØØØØ        DW      Ø,Ø,Ø
 D8AE Ø9DF              DW      DIRBUF
 D8BØ ØØØØ              DW      Ø
 D8B2 49DE              DW      CSV1
 D8B4 28DD              DW      ALV1

 D8B6 ØØØØ      DPTWO   DW      Ø
 D8B8 ØØØØØØØØØØ        DW      Ø,Ø,Ø
 D8BE Ø9DF              DW      DIRBUF
 D8CØ ØØØØ              DW      Ø
 D8C2 89DE              DW      CSV2
 D8C4 73DD              DW      ALV2

 D8C6 ØØØØ      DPTHRE  DW      Ø
 D8C8 ØØØØØØØØØØ        DW      Ø,Ø,Ø
 D8CE Ø9DF              DW      DIRBUF
 D8DØ ØØØØ              DW      Ø
 D8D2 C9DE              DW      CSV3
 D8D4 BEDD               DW      ALV3

                *****************************************************************
                *                                                               *
                * CBIOS RAM LOCATIONS THAT DON'T NEED INITIALIZATION.           *
                *                                                               *
                *****************************************************************

 D8D6 ØØ        CPMSEC  DB      Ø               ;CP/M SECTOR #
 D8D7 ØØ        CPMDRV  DB      Ø               ;CP/M DRIVE #
 D8D8 ØØ        CPMTRK  DB      Ø               ;CP/M TRACK #
 D8D9 ØØ        TRUESEC DB      Ø               ;DISK JOCKEY SECTOR THAT CONTAINS CP/M SECTOR
 D8DA ØØ        BUFDRV  DB      Ø               ;DRIVE THAT BUFFER BELONGS TO
 D8DB ØØ        BUFTRK  DB      Ø               ;TRACK THAT BUFFER BELONGS TO
 D8DC ØØ        BUFSEC  DB      Ø               ;SECTOR THAT BUFFER BELONGS TO
 D8DD           BUFFER  DS      1Ø24            ;MAXIMUM SIZE BUFFER FOR 1K SECTORS

 DCDD           ALVØ    DS      75              ;ALLOCATION VECTOR FOR DRIVE A
 DD28           ALV1    DS      75              ;ALLOCATION VECTOR FOR DRIVE B
 DD73           ALV2    DS      75              ;ALLOCATION VECTOR FOR DRIVE C
 DDBE           ALV3    DS      75              ;ALLOCATION VECTOR FOR DRIVE D
 DEØ9           CSVØ    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE A
 DE49           CSV1    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE B
 DE89           CSV2    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE C
 DEC9           CSV3    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE D
 DFØ9           DIRBUF  DS      128             ;DIRECTORY BUFFER

 DF89                   END
```

```
0006 AACK       000D ACR        0003 AETX       000A ALF        DCDD ALV0
DD28 ALV1       DD73 ALV2       DDBE ALV3       D3FA AUTOFLG    C500 BDOS
9000 BIAS       D300 BIOS       D8DA BUFDRV     0080 BUFF       D8DD BUFFER
D8DC BUFSEC     D8DB BUFTRK     D5D5 BUFWRTN    D3A0 CBOOT      BD00 CCP
0004 CDISK      D715 CICRT      D715 CIPTR      D692 CITBLE     E403 CITTY
D700 CIUC1      D715 CIUR1      D715 CIUR2      D3F2 CLDBOT     0019 CLEAR
D3FB CMNDBEG    D3FB CMNDEND    D6D2 COCRT      D6D6 COLPT      D64C CONIN
D652 CONIN1     D661 CONOUT     D640 CONST      D6D6 COPTP      D69A COTBLE
E406 COTTY      D6E1 COUL1      D6FF COUNT      D6D6 COUP1      D6D6 COUP2
D30C COUT       D5B5 CPMDMA     D8D7 CPMDRV     0016 CPMREV     D8D6 CPMSEC
D8D8 CPMTRK     D729 CSCRT      D729 CSPTR      D646 CSREADR    D6C2 CSRTBLE
D6BA CSTBLE     D721 CSTTY      D70C CSUC1      D729 CSUR1      D729 CSUR2
DE09 CSV0       DE49 CSV1       DE89 CSV2       DEC9 CSV3       D3F9 CWFLG
0008 DBLSID     DF09 DIRBUF     D583 DIVDONE    D57A DIVLOOP    E403 DJCIN
E406 DJCOUT     E42D DJDEN      E412 DJDMA      D333 DJDRV      E42A DJERR
E409 DJHOME     E400 DJRAM      E415 DJREAD     E40F DJSEC      E41B DJSEL
E430 DJSIDE     E427 DJSTAT     E40C DJTRK      E421 DJTSTAT    E418 DJWRITE
D886 DP1024D    D846 DP1024S    D856 DPB128D    D816 DPB128S    D866 DPB256D
D826 DPB256S    D876 DPB512D    D836 DPB512S    D8A6 DPONE      D8C6 DPTHRE
D8B6 DPTWO      D896 DPZERO     D590 DTSLOP     0005 ENTRY      D620 FILL
D5D4 FLUSH      D541 GETDPB     D3B3 GOCPM      D490 HOME       00C0 INTIOBY
D5C2 INTO       0003 IOBYTE     D681 LIST       D684 LIST1      D68C LISTST
D732 LSLPT      D6CA LSTBLE     D6A2 LTBLE      0004 MAXDISK    D393 MESSAGE
D59F MOVE       D635 MOVER      D637 MOVLOP     E800 MSDV       0038 MSIZE
D45E NEWDMA     D43F NEWSEC     D456 NOWRAP     E000 ORIGIN     D5BD OUTOF
D67C PNCH1      D5DB PREP       D336 PROMPT     D6AA PTBLE      D676 PUNCH
D6F6 PWAIT      D5B8 RDWR       D66C READER     D569 READ       D66F READERA
D672 READR1     D737 READY      D56D REDWRT     000A RETRIES    D5E4 RETRYLP
D612 RETRYOP    001E REVNUM     D6B2 RTBLE      D5A4 SECPSEC    D56E SECSIZ
D497 SECTRAN    D656 SELDEV     D48A SETDMA     D4C6 SETDRV     D524 SETDRV1
D485 SETSEC     D492 SETTRK     D4A5 SIDEA      D517 SIDEOK     D4A8 SIDEONE
D4AE SIDETWO    D724 STAT       D73A TINIT      0100 TPA        D8D9 TRUESEC
D43E WARMLOD    D472 WARMRD     D303 WBOOTE     D3FC WBOOT      0000 WBOT
D400 WFLG       D562 WRITE      D5CC WRITTYP    D475 WRMREAD    D7D5 XLT124
D748 XLT128     D763 XLT256     D798 XLT512     D55A XLTS       D53D ZRET
```